Interoperability of Reconfiguring System on FPGA Using a Design Entry of Hardware Description Language

Ferry Wahyu Wibowo[1]
[1] Department of Informatics Engineering, STMIK AMIKOM Yogyakarta, Yogyakarta, Indonesia
Email: ferrywahyu@gmail.com

***Abstract*—For a long ago, world of digital design has spread out in the many major and a lot of logics, approaches, and theories has been proposed. The digital emerges as a solution of a daily-life need and applicable on such technology from the developing devices until software-based. All of the designs has a significant point on the spesification, integration, and optimization. The designers have been trying to make a good designs on both hardware and software, latest both combinations have been known as the basic idea of hardware/ software co-design. The state-of-the art computer is very interesting to research because of its implementation can make changes of the cycle of reconfigurable objects. This paper presents a comparison of the two role plays in reconfigurable devices especially FPGA-based, i.e. Altera and Xilinx. The idea is that of a simple compiler has a good performance designs for synthesizing Very high speed integrated circuit Hardware Description Language (VHDL) code as well as the other complexity software that more powerful. So, this paper proposes such method as interoperability for reconfiguring devices to get the point why few of the standard VHDL code can't be synthesised in the different compiler of VHDL code between Xilinx and Altera. The project of compiler softwares that is observed from Xilinx is ISE and from Altera is Max+Plus II. Max+Plus II is a low-cost software than ISE Xilinx, although both Xilinx and Altera devices have a different structure each other.**

***Keywords*—Altera, FPGA, Interoperability, Reconfiguring, VHDL, Xilinx**

I. Introduction

While early Field Programmable Gate Arrays (FPGAs) have been introduced that are the standing devices of reconfigurable computing platform to reach widely of very large scale integrated (VLSI) technology. Emerging FPGAs has driven some issues from ideas until how to get integrated board on one piece that is capable to be efforted and designed. Now, the need of complexity is possible to do and lacks of designing are always observing in a good maner. We realize that all of the hardware designers, especially who have been working on Field Programmable Gate Arrays (FPGAs) or Application Specific Integrated Circuits (ASICs), are familiar with how to design them and making some comparisons of each of the designing tools. In the field of computing machines, the target is to make a good combination between both application-hardware and driven-software. A software designers have to be familiar with hardware design to work on such technology applications [1]. Some of the designers hide their designs to the other designers to make differ on their intellectual properties and the other reason is they do not want to be revealed their ideas in such technology.

FPGAs as a high density and high-performance embedded system design make an ease-of-use to configure a brand technology of dynamic reconfiguration. Nowadays, developing FPGAs technology is very rapidly due to time-to-market for large design. A static random access memory (SRAM) FPGAs as a part of FPGAs components are vulnerable to be cracked by bitstream cloning, reverse-engineering and tampering [2]. FPGAs compiler is the solution to optimize such a design to get hardware platform. This paper presents of the big of two vendors that play on the role of reconfigurable devices, i.e. Xilinx and Altera. Xilinx has been developing ISE tool to compile its hardware description language design, and the otherside, Altera has developed Max+Plus II, nevertheless Altera has been developing noval software product that is Quartus II. But, this paper is not going to explain further about the term of Quartus II, it just investigates the difference of compiling methods between both Altera Max+Plus II and Xilinx ISE 9.2i on the designs although they use a same standard hardware description language (HDL). A hardware description language that is observed in this paper is Very high speed integrated circuit Hardware Description Language (VHDL). ISE and Quartus II softwares have a bigger capasity than Max+Plus II, but the capasity of ISE software is biggest. This paper also proposes an idea using Max+Plus II to design a hardware projects than using expensive tools, because of training or education in the area of reconfigurable computing is very low-need to be understood in short-time and in order to get low-cost designing and compiling on FPGAs, although we knew that each of vendors has a different methods on their compiler and reconfigurable device platforms. The goal is to maximize the use of a low-cost tool to find a good performance on its implementation. This paper presents how the same entry design of VHDL code can be compiled and ran on both Xilinx and Altera FPGAs with some modifications on the programming of VHDL code, because of each vendor had developed an own-libraries on theirs tools, therefore analyzing on the standard VHDL code and errors have been observed to make a designs as well.

*Corresponding author, Tel.: +628157948404
E-mail address: ferrywahyu@gmail.com (F.W. Wibowo).

DOI: 01.IJIT.02.01.62

II. RELATED WORKS

A. Over view of VHDL

A VHDL was developed as a mandate of Department of Defense (DoD) for federally-sponsored VLSI designs in 1980s. The first convention established during the purpose of novel technology of reconfigurable chip design as The Institute of Electrical and Electronics Engineer (IEEE) standard, IEEE 1076, in 1987 and became extended convention in IEEE standard, IEEE 1164, in 1993 and during 1996, IEEE 1076.3 became a VHDL synthesis standard. VHDL stands for Very high speed integrated circuit Hardware Description Language. VHDL is Pascal-like but for creating an electronic systems and reconfiguring hardware like ASICs or FPGAs. The differences of both the hardware and software are, a software is a bundle of statements which are executed sequentially while a hardware has an event that happens concurrently and the execution of hardware is called synthesis. A software language can't be used for describing and simulating hardware, and either the concurrent software languages can't be used. VHDL supports the development, verification, synthesis, and testing of hardware designs; the communication of hardware design data; and the maintenance, modification and procurement of hardware. A VHDL has been corrected and clarificated due to its standard until replaced by subsequent document or until the standard is officially revised [3]. We use VHDL code to write a program and describe hardware based on digital electronics circuit or a chip just like a schematic does, but the difference on its design-use is, a circuits are very complex to be designed by schematics.

A VHDL program is a collection of modules that form a hierarchical design which we have known as a top-down design. From the top-down design, we are capable to describe how to design our circuits and we may get a synthesized circuit with a same consumption of components of FPGAs using other algorithms constructed and designed by VHDL code [4]. The main format to design using VHDL code consists of three parts, i.e. library, design entity, and design architecture. The design entity is the primary hardware abstraction and representing the inputs and outputs of the whole design of system. The design architecture is the internal description of implementation that relates on the signal processing and controlling and can be written in one of three different detail levels as structural, or dataflow, or behavioral. Occasionally, each level of designs can be made a hybrid designs to build a big picture of such implementation. The designs have a trade-off and self-characterization that of bringing other-side of developing a concept of a systems or implementations. A structural design using explicit components and the connections between them are defined, and a dataflow design using most statements is assigning expressions to signal, in this way the tools are heavily involved in converting the text to hardware. Meanwhile behavioral design using an algorithm that of describing the circuit's output which is developed; in this design level, the tools may not be able to convert the text to hardware and may not be synthesizable and if could, it may lead to a very large circuit and has a slow or inefficient realization in implementation but primarily fine used for simulation and verification.

There is no international convention how a VHDL-compliant tool must behave. So, that made a vendors built standarizations of theirs tools to synthesis theirs reconfigurable devices. However, when we are working on VHDL compiler, it analyzes VHDL code for syntax errors and checks for compatibility with other modules. VHDL compiler generated an information about the project that we synthesis to keep track of definitions via entity and architecture names. It also contains analysis results about the system on FPGAs. The running steps are very easily to understand, the synthesizer converts a VHDL program into circuit with components that can be found in the netlist result, then the compiler executes for placing and routing to fit the circuit to a die (see fig. 1).

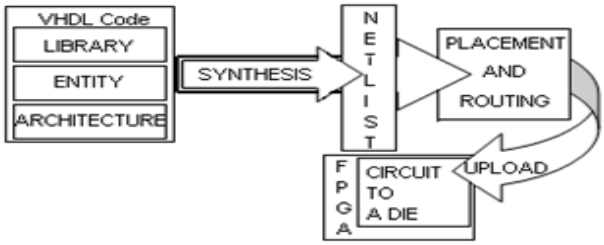

Figure 1. Running on VHDL compiler

All vendors has same an idea in that of building theirs compiler tools, but in how to implement the synthesising of theirs design entries embeds on FPGAs they has own-standardization. However, for our designs we can choose whatever tools which are very familiar with. But, this paper explores a low-cost tool that is capable to interoperate on the design with the complex tool as mention previously, nevertheless this paper ignores the devices we used.

III. EXPERIMENT SUPPORTS AND TOOLS

A. Xilinx ISE

ISE stands for Integrated Synthesis Environment that was found by Xilinx. ISE Xilinx tools generate lines of VHDL code automatically in the file to construct a starting circuit input definition. The generated code includes library definitions, an entity statement, an architecture statement with begin and end statements, and a comment block template for documentation. Due to the richness of the VHDL code, there are many different ways to define a circuit inputs inside of a VHDL test bench module [5]. ISE Xilinx is not stand-alone software to get a synthesising, an analyzing, and an uploading on FPGAs, but it has related with 3rd party partner tools. To develop its tools, Xilinx has a program Xilinx University Program (XUP). Basically, XUP aims at a world-wide program to encourage and help the academic communities to use Xilinx technologies and make an easy access to Xilinx best-in-class development tools and hardware platforms. The design entries of ISE tools are more complex every time including HDL edit and entry, system generator for digital signal processing (DSP), intellectual property (IP) core generator, architecture wizards, ECS schematic editor and register-transfer-level

DOI: 01.IJIT.02.01.62

(RTL) checker. Synthesis technology that is used by Xilinx ISE is xilinx synthesis technology (XST). The technologies for design verification are ModelSim Xilinx Edition, Static Timing Analyzer, ChipScope Pro, Xpower estimation, ChipViewer, FPGA Editor with Probe, and HDL Bencher testbench generator. Xilinx also uses board level integration that is I/O buffer information specification (IBIS) models. In order to get the powerful implementation analyzing on FPGAs, Xilinx embedded its tools by floorplanner and pinout and area constraints editor (PACE), constraint editor, timing driven place & route, modular design, incremental design, timing improvement wizard [6]. Not all constraints can be specified with all tools or methods, if a tool or method is not listed for that constraint, the reconfigurable device designers cannot use the constraint with it [7]. Xilinx always develops its softwares to optimaze designing from beginning as resumed in table I.

TABLE I. THE DEVELOPMENT OF THE XILINX SOFTWARES

No.	Issues	Year
1.	Xilinx increased the effectiveness of its tools of DS23 Automated Design Implementation (ADI).	1989
2.	Xilinx performs ADI upgrade, version 3.0.	1990
3.	X-BLOX provides high-level schematic entry and finite states using Xilinx ABEL.	1992
4.	Xilinx had merged with NeoCAD onto development system technologies for implementing design XC3000 and XC4000 FPGA families.	1995
5.	Xilinx had completed its software with combination of Windows-based design tools, integrating industry-standard HDL, synthesis, schematic entry and simulation tools with the Xilinx XACTstep[TM] implementation tools.	1996
6.	Teams of software engineers from Synopsys, Sinplicity, Model Technology, Visual Software Solutions, and Xilinx have created the ultimate in design automation tools-Xilinx ISE that will manage all modules in the design.	2000
7.	Xilinx introduces fast, effisient ISE 4.1i software.	2001
8.	New release of Xilinx ISE 5.1 that designs with 15% higher-performance (2x faster than ISE 4.1i) – allowing the use of slower devices to achieve a cost savings of at least one device speed grade.	2002
9.	New ISE 8.1i software is operable being faster timing closure.	2005

The ISE tools are more powerful to design and user friendly with FPGA designers, beside that Xilinx always develops its tool features until the FPGA designers are very enjoyable with the tool. It has a slogan "one solution for all devices and all your logic design needs".

B. Altera Max+Plus II

Max+Plus stands for Multiple Array MatriX Programmable Logic User System, that was found by Altera. Altera Max+Plus II software can be installed on Personal Computers (PCs) and UNIX workstations. Altera also provides a multi platform, architecture-independent design environment that is very easily adaptation a specific design needs. Max+Plus II offers easy design entry, quick processing, and straightforward device programming. Max+Plus II development software is fully integrated package for creating logic designs of Altera programmable logic devices-including the Classic, MAX 5000, MAX 7000, MAX 9000, FLEX 6000, FLEX 8000, and FLEX 10K families of devices. Max+Plus II offers a full spectrum of logic design capabilities: a variety of design entry methods for hierarchical designs, powerful logic synthesis, timing-driven compilation, partitioning, functional and timing simulation, linked multi-device simulation, timing analysis, automatic error location, and device programming and verification. Max+Plus II also reads Xilinx netlist files and writes standard delay format (SDF) files for a convenient interface to other industry standard computer-added-engineering (CAE) software. The design editors (the graphic, text, and waveform editors) and auxiliary editors (the floorplan and symbol editors) also share numerous features. Each design editor allows the designer to perform similar task-such as finding a signal or symbol-in the same way [8].

The Max+Plus II compiler consists of a series of modules and a utility that check a project for errors, synthesize the logic, fit the project into one or more Altera devices, and generate output files for simulation, timing analysis, and device programming. The Max+Plus II compiler is providing powerful project prosessing that is customizable to achieve the best possible silicon implementation. The superb integration of the Max+Plus II software helps the designer to maximize his efficiency and productivity. The compilation process steps of Max+Plus II can be explained as the compiler first extracts information that defines the hierarchical connections between a project's design file and checking the project for basic design entry error. It creates an organizational map of the project and then combines all design files into a fully flattened database that can be processed efficiently. The compiler also creates programming files which the Max+Plus II programmer or another industry-standard programmer defines to program one or more Altera devices [8].

IV. ANALYSIZES AND RESULTS

In 2004, Altera had analyzed its software, Quartus II versus Xilinx ISE sofware [9]. Some features and comparisons had been published which target was to ask the FPGA designers migrate its devices design expertise from Xilinx ISE into Altera Quartus II, or at least, using the features of its tool to do working on the Altera devices. However, those are not our main discussion. This paper investigates some designs using VHDL code on both Altera Max+Plus II 10.2 and Xilinx ISE 9.2i and also discusses how to get a specific designs of VHDL code that can be compiled on both tools because there are some designs must use a tricky ways to get a solution of designing of VHDL code. The conversion of designs from Max+Plus II to Xilinx ISE and the other way seems to be easy if the circuit designers use standard VHDL and many libraries are only defined by such tools. But, for several VHDL program can't act like that although the circuit designer has use the standard VHDL. It seems such VHDL compiler have a specific algorithm that was embedded.

The simple design of standard VHDL should be capable to define in any other VHDL compiler. The VHDL program for

DOI: 01.IJIT.02.01.62

data-flow design can be defined as shown below :

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  entity fully is
4  port(a, b, c : in std_logic;
5        sum, carry : out std_logic);
6  end fully;
7  architecture paper of fully is
8  begin
9       sum <= (a xor b) xor c;
10      carry <= (a and b) or (a and c) or (b and c);
11 end;
```

The circuit design of fully can be compiled on both Xilinx ISE 9.2i and Altera Max+Plus II succesfully, because of the design is defined in standard VHDL, but when the design of VHDL program is combined with a same circuit design, or otherwise the design is defined in a structural method then the design is resulted as a circuit of the VHDL code as below :

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4  entity fully2 is
5  port (A, B : in unsigned(1 downto 0);
6  C : out unsigned(2 downto 0));
7  end;
8  architecture paper of fully2 is
9  component fully
10 port (a, b, c : in std_logic;
11 sum, carry : out std_logic);
12 end component;
13 signal carry : std_logic;
14 begin
15 bit0 : fully port map (
16   a=>A(0),b=>B(0),c=>'0',sum=>C(0),carry=>carry);
17 bit1 : fully port map (
18 a=>A(1),b=>B(1),c=>carry,sum=>C(1),carry=>C(2));
19 end;
```

The synthesis can be done by Xilinx ISE but not by Altera Max+Plus II. First result states that Altera Max+Plus II can't define ieee.numeric_std.all, so it shoud be replaced and defined with the library using ieee.std_logic_arith.all. But, there is still a problem because of the Altera Max+Plus II compiler can not define unsigned as in line 5 and 6, so the description should define the VHDL standard into std_logic for single input or output; or std_logic_vector for multi inputs or outputs. Other problem emerges due to definition in the line 16 is '0' valued by c. The solution for the value of c can be defined as a signal, and the other problem is the Altera Max+Plus II do not support source that is reused by VHDL program as signal carry of fully2. This solution can be prevented by writing a simple label straightforward without symbol '=>'. The difference of VHDL code implementation on both ISE Xilinx and Max+Plus II Altera compilers, even they have structured syntaxes of a standard VHDL code, is always related with the methods of VHDL compiler that are used by vendors. The circuit designers who have familiar with a specific methods and implementations of VHDL code, although they have expertise in that area, they have to understand the characteristics of tools and devices used. The standard VHDL works on the domain of syntaxes but not for such coding or reconfiguring in tools and devices. The specific warning and notification in the VHDL code compiler are always developed and found by vendors which have feedback of VHDL programming and FPGAs reconfiguring from the circuit designers. A data-flow method for multiplexer on the concurrent design of VHDL code can be written as below:

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_arith.all;
4  entity multi is
5  port(in0, in1, in2, in3 : in std_logic_vector(15 downto 0);
6       s : in std_logic_vector(1 downto 0);
7       z : out unsigned(15 downto 0));
8  end;
9  architecture paper of multi is
10 begin
11 z <= in0 when s = "00" else
12      in1 when s = "01" else
13      in2 when s = "10" else
14      in3 when s = "11" else (others => 'X');
15 end exam;
```

For the synthesising of the data-flow design of multiplexer on both the Xilinx ISE and Altera Max+Plus II is successfully generated, but, because of the capasity constraint of Altera FPGAs, so the synthesis result needs more consumption of FPGAs components. But synthesising for Xilinx devices consumes a small part of components. In the modelling of such circuit system, the synthesis result of a large input and output is usable in the sub-module. A good design of using VHDL code is more efficient and effective, this can be done by knowing concept of digital circuit, not just knowing how to program with VHDL code or other HDL. The behavioral method of circuit design need to be analyzed more than other designs, because of the consumption of FPGAs components is very large consumed than other methods of circuit designs when it has synthesised. An example of process method can be described as below :

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_arith.all;
4  entity counter is
5  port(Clk, Reset : in std_logic;
6       Q : out unsigned(3 downto 0));
7  end counter;
8  architecture exam of counter is
9  signal count : unsigned(3 downto 0);
10 begin
11 process (Clk)
12 begin
13 if rising_edge(Clk) then
14       if Reset = '1' then count <= (others => '0');
15          else count <= count + 1;
16       end if;
```


DOI: 01.IJIT.02.01.62

```
17 end if;
18 end process;
19 Q <= count;
20 end;
```

The VHDL code design of counter can be synthesised both in Altera Max+Plus II and Xilinx ISE 9.2i. There is a different definition for its coding than mentioned before, that signed by using data type of unsigned. The sequentially design that is written by programming of VHDL code has different characteristics when the circuit design is applicable on ISE Xilinx and Altera Max+Plus II, it seems more succesfully compiled using Xilinx ISE, but it can be written in the Altera Max+Plus II by separating sequentially programming in the sequentially programming. So, the result of synthesising a kind of circuit design can be defined clearly and not making a compiler defines a multi statements. This method also can make consuming of FPGAs components that is defined in Altera Max+Plus has large consumption than defining in the ISE Xilinx.

Conclusions

The software development is always related with hardware design in reconfigurable computing. We can't deny that both the hardware and software can't be ignored if we want to design a large systems. A VHDL has been standarized by IEEE but in fact each vendor has a standarization of theirs devices and tools. This paper has discussed how to implement a good design using VHDL code from the tool which has defined to configure an FPGAs and some tricky ways has been presented. Perhaps, for the advanced designers that is not matter because they has some costs to buy, but for people who hasn't, they have to recognize their designs.

Acknowledgment

I thank to the God who gives me chance to do this research. I also thank to Prof. Dr. Jazi Eko Istiyanto who has encouraged me to be involved in the major of FPGAs researches, until i can do my own-researches.

References

[1] A. Virginia, Y.D. Yankova, K Bertels, "An empirical comparison of ANSI-C to VHDL compilers: SPARK, ROCCC and DWARV," *Annual Workshop on Circuits, Systems and Signal Processing*, ProRISC 2007, pp. 388 - 394, Utrecht, 2007.

[2] A.S. Zeineddini, K. Gaj, "Secure Partial Reconfiguration of FPGAs," *Proceedings 2005 IEEE International Conference on Field Programmable Technology*, pp. 155-162, 2005.

[3] IEEE Std 1076, "*IEEE Standard VHDL Language Reference Manual*," The Institue of Electrical and Electronics Engineers, Inc., 2000.

[4] F.W. Wibowo, "The Conservative Structure of Synthesizing Read Only Memory Design using VHDL on FPGA," *International Seminar on Industrial Engineering and Management, 4th*, pp. 231-235. 2010.

[5] Digilent, *Xilinx ISE Simulator (ISim) VHDL Test Bench Tutorial*, Digilent, Inc., February 2010.

[6] M. Crastes, *FPGAs in Education*, Xilinx Inc., 2007.

[7] Xilinx, *XST User Guide for Virtex-6, Spartan-6, and 7 Series Devices*, Xilinx Inc., July 2011.

[8] Altera, *Max+Plus II Getting Started*, Altera Corporation, 1997.

[9] Altera, "Advantages of Quartus II Software over Xilinx ISE," *White Paper*, Altera Corporation, 2004.

DOI: 01.IJIT.02.01.62